Section 6: Cross-Validation

Yotam Shem-Tov

Fall 2015

In Sample prediction error

- There are two types of Prediction errors: In sample prediction error and out of sample prediction error.
- In sample prediction error: how well does the model explain the data which is used in order to estimate the model.
- Consider a sample, (y, X), and fit a model $f(\cdot)$ (for example a regression model), and denote the fitted values by $\hat{y}_i$.
- In order to determine how well the model fits the data, we need to choose some criterion, which is called the loss function, i.e $L(y_i, \hat{y}_i)$.
- standard loss functions:
  $MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$, $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$

Out of sample prediction error

- How well can the model predict a value of y_j given x_j where observation j is not in the sample. This is referred to as the out of sample prediction error.
- How can we estimate the out of sample prediction error?
- The most commonly used method is **Cross-Validation**.

Cross-Validation

Summary of the approach:

1. Split the data into a training set and a test set
2. Build a model on the training data
3. Evaluate on the test set
4. Repeat and average the estimated errors

Cross-Validation is used for:

1. Choosing model parameters
2. Model selection
3. Picking which variables to include in the model

Cross-Validation

There are 3 common CV methods, in all of them there is a trade-off between the bias and variance of the estimator.

1. Random sub-sampling CV
2. K-fold CV
3. Leave one out CV (LOOCV)

My preferred method is *Random sub-sampling CV*.

Random sub-sampling CV

1. Randomly split the data into a test set and training set.
2. Fit the model using the training set, *without using the test set at all!*
3. Evaluate the model using the test set
4. Repeat the procedure multiple times and average the estimated errors (RMSE)

What is the tuning parameter in this procedure?
The *fraction* of the data which is used as a test set There is no common choice of *fraction* to use. My preferred choice is 50%, however this is arbitrary.

Predicting union membership

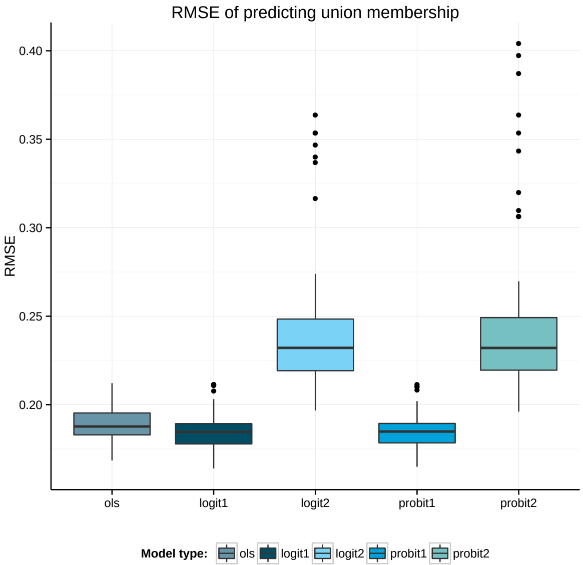
- We will use PSID data (from the AER package) and try to predict union membership.
- We will look at 5 different models: OLS, Logit with main effects, Logit with all two way interactions, Probit with main effects, Probit with all two way interactions.
- I used a fraction of 50% as the test set and 50% as the training set.
- What is your guess, which model will perform best?
- The results are:

```
> tapply(dp$rmse,dp$model,mean)
      ols    logit1    logit2   probit1   probit2
0.1892946 0.1849251 0.2434963 0.1854004 0.2480870
```

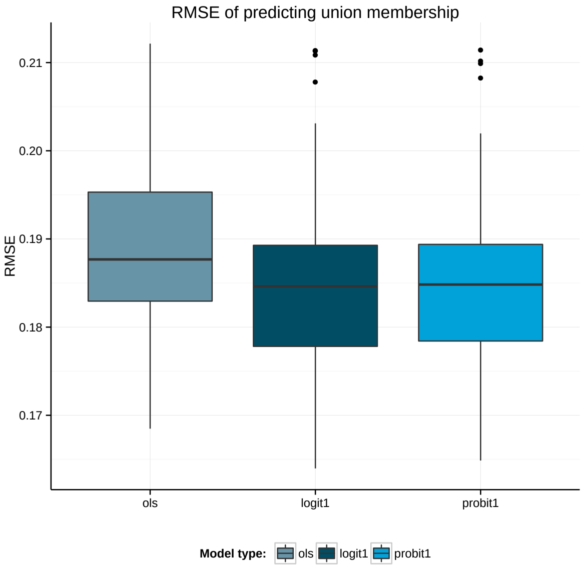
Predicting union membership: Split-sample CV

```
L0=100 # number of repetitions
rmse.ols <- rmse.probit1 <- rmse.probit2<-rmse.logit1 <- rmse.logit2 <-rep(NA,L0)
data = d[,c(covnames,"tr")]
for (j in c(1:L0)){

id = sample(c(1:dim(d)[1]),round(dim(d)[1]*0.5))

ols <- lm(tr˜(.),data=data[id,])
logit1 <- glm(tr˜(.),data=data[id,],family=binomial(link="logit"))
logit2 <- glm(tr˜(.)ˆ2,data=data[id,],family=binomial(link="logit"))
probit1 <- glm(tr˜(.),data=data[id,],family=binomial(link="probit"))
probit2 <- glm(tr˜(.)ˆ2,data=data[id,],family=binomial(link="probit"))

rmse.ols[j] = rmse(predict(ols,newdata=data[-id,],type="response"),d$tr[-id])
rmse.logit1[j] = rmse(predict(logit1,newdata=data[-id,],type="response"),d$tr[-id])
rmse.logit2[j] = rmse(predict(logit2,newdata=data[-id,],type="response"),d$tr[-id])
rmse.probit1[j] = rmse(predict(probit1,newdata=data[-id,],type="response"),d$tr[-id
rmse.probit2[j] = rmse(predict(probit2,newdata=data[-id,],type="response"),d$tr[-id

}
```

Random sub-sampling CV: Predicting union membership



Random sub-sampling CV: Predicting union membership

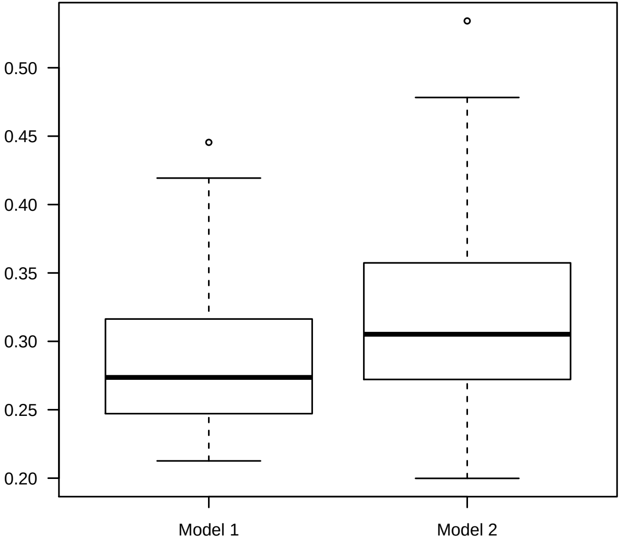


Random sub-sampling CV: P-score Welder Example

We can use CV in order to choose between the two competing models we discussed in the Welder example.

```
 L0=100 # number of repetitions
rmse.model.1 <- rmse.model.2 <- rep(NA,L0)
a = data.frame(treat=treat,x)
for (j in c(1:L0)){
  id = sample(c(1:dim(d)[1]),round(dim(d)[1]*0.5))

  ps.model1 <- glm(treat~(.),data=a[id,],family=binomial(link=
  ps.model2 <- glm(treat~(.)^2,data=a[id,],family=binomial(lin

  rmse.model.1[j]=rmse(predict(ps.model1,newdata=a[-id,],
  type="response"),a$treat[-id])
  rmse.model.2[j]=rmse(predict(ps.model2,newdata=a[-id,],
  type="response"),  a$treat[-id])
}
```

Random sub-sampling CV: P-score Welder Example



Random sub-sampling CV: P-score Welder Example

- The results are in the table below:

	Model 1	Model 2
Mean	0.29	0.33
Median	0.29	0.32

- It is clear that model 1, no interactions, has a lower out of sample prediction error.
- Model 2 (with interactions) over fits the data, and generates a model with a wrong P-score. The model includes too many covariates
- Note, it is also possible to examine other models that include some of the interactions, but not all of them

K Folds CV

- Randomly split the data into K folds (groups)
- Estimate the model using $K-1$ folds
- Evaluate the model using the remaining fold.
- Repeat the process by the number of folds, K times
- Average the estimated errors across folds

The choice of K, is a classic problem of bias-variance trade-off.

What is the tuning parameter in this method? The *number of folds*, K.
There is no common choice of K to use.
Commonly used choices are, $K = 10$, and $K = 20$. The choice of K depends on the size of the sample, N.

K Folds CV: union membership

Next we return to the union membership problem, but using K-folds CV. Write code to implement a K-folds CV procedure:

```
folds.num <- 10
d$fold <- cut(c(1:dim(d)[1]),breaks=folds.num)
levels(d$fold) = c(1:folds.num)

rmse.ols <- rmse.probit1 <- rmse.probit2<-rmse.logit1 <- rmse.logit2 <- converge1<-
data = d[,c(covnames,"tr")]
for (j in c(1:folds.num)){
  id = which(d$fold!=j)

  ols <- lm(tr˜(.),data=data[id,])
  logit1 <- glm(tr˜(.),data=data[id,],family=binomial(link="logit"))
  logit2 <- glm(tr˜(.)ˆ2,data=data[id,],family=binomial(link="logit"))
  probit1 <- glm(tr˜(.),data=data[id,],family=binomial(link="probit"))
  probit2 <- glm(tr˜(.)ˆ2,data=data[id,],family=binomial(link="probit"))

  rmse.ols[j] = rmse(predict(ols,newdata=data[-id,],type="response"),d$tr[-id])
  rmse.logit1[j] = rmse(predict(logit1,newdata=data[-id,],type="response"),d$tr[-id
  rmse.logit2[j] = rmse(predict(logit2,newdata=data[-id,],type="response"),d$tr[-id
  rmse.probit1[j] = rmse(predict(probit1,newdata=data[-id,],type="response"),d$tr[-
  rmse.probit2[j] = rmse(predict(probit2,newdata=data[-id,],type="response"),d$tr[-
}
```

K Folds CV: union membership

The results are:

```
> tapply(dp$rmse,dp$model,mean)
      ols    logit1    logit2   probit1   probit2
0.1851554 0.1806641 0.1946168 0.1812608 0.1951449
```

Is it always true that K-folds CV and Split-sample CV will yield the same result? No.

The tuning parameter

- K folds,

Choosing the number of folds, K

$\uparrow K$ lower bias, higher variance
$\downarrow K$ higher bias, lower variance

- Random sub-sampling,

Choosing the fraction of the data in the test set

$\downarrow$ *fraction* lower bias, higher variance
$\uparrow$ *fraction* higher bias, lower variance

Leave one out CV (LOOCV)

- LOOCV is a specific case of K folds CV, where $K = N$

- Example in which there is an analytical formula for the LOOCV statistic
- The model: $Y = X\beta + \varepsilon$
- The OLS estimator: $\hat{\beta} = (X'X)^{-1} X'y$
- Define the hat matrix as, $H = X(X'X)^{-1} X'$
- Denote the elements on the diagonal of H, as h_i
- The LOOCV statistic is,

$$CV = \frac{1}{n} \sum_{i=1}^{n} (e_i/(1 - h_i))^2$$

where $e_i = y_i - x_i' \hat{\beta}$, and $\hat{\beta}$ is the OLS estimator over the whole sample

A classic example for using CV

- A classic use of CV procedures is for choosing a nuisance parameter.

Definition

In statistics, a nuisance parameter is any parameter which is not of immediate interest but which must be accounted for in the analysis of those parameters which are of interest.

- Example: Regression shrinking and selection via LASSO, see Tibshirani (1996) .

$$(\beta, \lambda) = \underset{\beta,\lambda}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{k=1}^{K} \beta_k x_{ki} \right)^2 + \lambda \sum_{k=1}^{K} |\beta_k|$$

This problem is equivalent to,

$$(\beta, \lambda) = \underset{\beta,\lambda}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{k=1}^{K} \beta_k x_{ki} \right)^2 \quad \text{subject to} \sum_{k=1}^{K} |\beta_k| \leq t$$

- The parameter λ (or t) is a nuisance parameter that we can use a CV procedure to choose its value.

CV in time series data

- The CV methods discussed so far do not work when dealing with time series data
- The dependence across observations generates a structure in the data, which will be violated by a random split of the data
- Solutions:
  1. An iterated approach of CV
  2. Bootstrap 0.632 (?)

CV in time series data

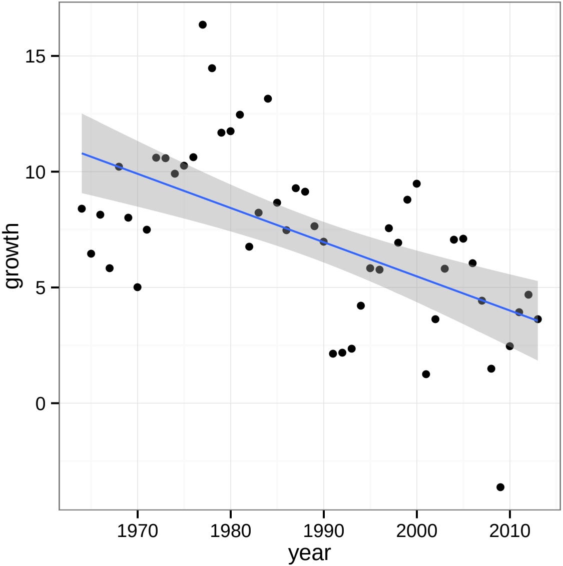
Summary of the iterated approach:

1. Build a model using the first M periods
2. Evaluate the model on period $t = (M + 1) : T$
3. Build a model using the first $M + 1$ periods
4. Evaluate the model on period $t = (M + 2) : T$
5. Continue iterating forward until, $M + 1 = T$
6. Average over the estimated errors

Example

- We want to predict the GDP growth rate in California in 2014
- The available data is *only* the growth rates in in the years 1964 – 2013
- consider the following three possible Auto-regression models:
  1. $y_t = \alpha + \beta_1 y_{t-1}$
  2. $y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-2}$
  3. $y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \beta_3 y_{t-3}$

Example: The data



Example: estimation of the three models

	Model 1	Model 2	Model 3
Intercept	1.954^{*}	1.935^{*}	1.411
	(0.841)	(0.919)	(0.977)
Lag 1	0.717^{***}	0.710^{***}	0.716^{***}
	(0.103)	(0.149)	(0.149)
Lag 2		0.014	-0.145
		(0.150)	(0.182)
Lag 3			0.217
			(0.150)
R^2	0.505	0.509	0.534
Adj. R^2	0.495	0.487	0.502
Num. obs.	49	48	47

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Example: choice of model

- Which of the models will you choose?
- Will you use an F-test?
- What is your guess: which of the models will have a lower *out of sample error*, using CV?

Example: F-test I

- Note, in order to conduct an F-test, we need to drop the first 3 observations. This is in order to have the same data used in the estimation of all three models.
- Dropping the first 3 observations, might biased our results in favour of models 2 and 3, relative to model 1.

```
Analysis of Variance Table

Model 1: y ~ lag1 + lag2
Model 2: y ~ lag1 + lag2 + lag3
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     44 330.02
2     43 314.58  1    15.438 2.1102 0.1536
```

Example: F-test II

```
Analysis of Variance Table

Model 1: y ~ lag1
Model 2: y ~ lag1 + lag2
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     45 330.03
2     44 330.02  1  0.012439 0.0017 0.9677
```

Example: F-test III

```
Analysis of Variance Table

Model 1: y ~ lag1
Model 2: y ~ lag1 + lag2 + lag3
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     45 330.03
2     43 314.58  2     15.45 1.0559 0.3567
```

Example: CV Results

- We used the iterative approach, as this is time series data
- M is the number of periods used for fitting the model before starting the CV procedure.
- The average $RMSE$ are,

	Model 1	Model 2	Model 3
$M = 5$	27.266	27.078	26.994
$M = 10$	29.770	29.586	29.474
$M = 15$	33.106	32.924	32.797

- Among *Model* 1 and *Model* 2 only, which is preferable?

The tuning parameter in time series CV

- What is the bias-variance trade-off in the choice of M?

Choice of M

$\uparrow M$ lower bias, higher variance

$\downarrow M$ higher bias, lower variance

Additional readings

- For a survey of cross-validation results, see Arlot and Celisse (2010), http://projecteuclid.org/euclid.ssu/1268143839